Campus Monterrey

Inteligencia artificial avanzada para la ciencia de datos II (Gpo 501)

Proyecto NLP: Síntesis y transcripción de un archivo de audio.

Juan Pablo Castañeda Serrano A01752030

Overview

La interfaz, y el código de este se encuentran aqui:
https://github.com/a01752030/NLP_Proyecto

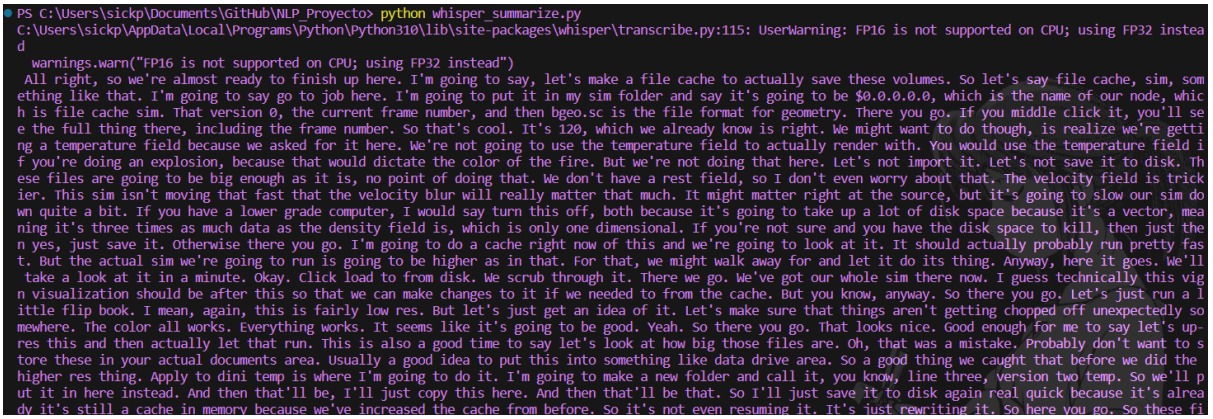
Este proyecto se divide en 2 secciones y utiliza varios recursos para su realización. Empezare por explicar cada una de estas secciones, de modo que pueda explicar las tecnologías utilizadas sobre la marcha

Archivo de interfaz en consola

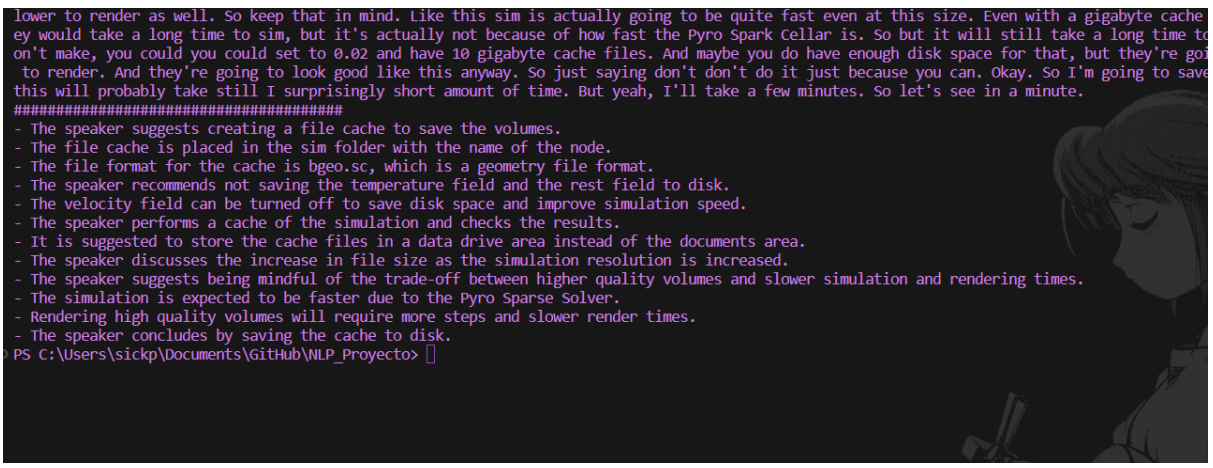
Este script tiene como objetivo principal procesar un archivo de audio mediante el uso del modelo de reconocimiento de voz "Whisper". Primero, intenta cargar dicho modelo y, si tiene éxito, procede a transcribir el contenido de un archivo de audio específico ('MA2.m4a'). La transcripción resultante se imprime en la consola. Posteriormente, utiliza la transcripción como entrada para una conversación simulada con ChatGPT, donde se establece que el usuario es un administrador de oficina y se le solicita resumir el texto en puntos clave. La respuesta generada por ChatGPT se imprime también en la consola. En caso de que ocurra algún error durante la carga del modelo Whisper, la transcripción del audio o la interacción con ChatGPT, se emiten mensajes de error correspondientes.

Ejecución de código de consola:

- Programa y transcripción



- Síntesis

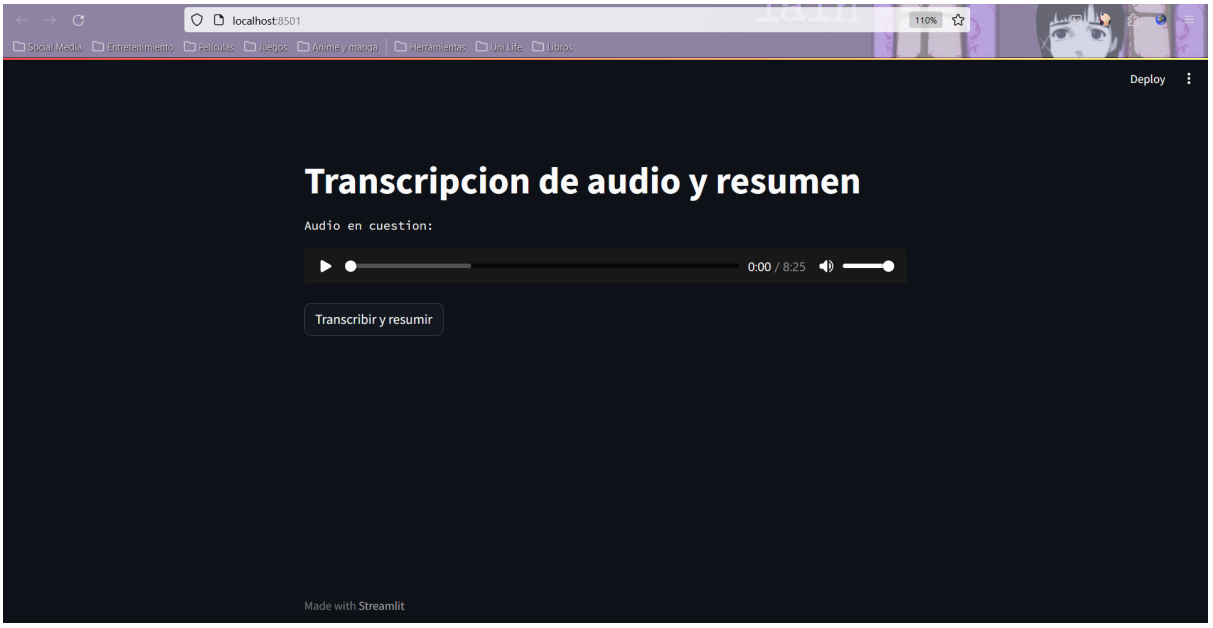


Archivo de interfaz gráfica

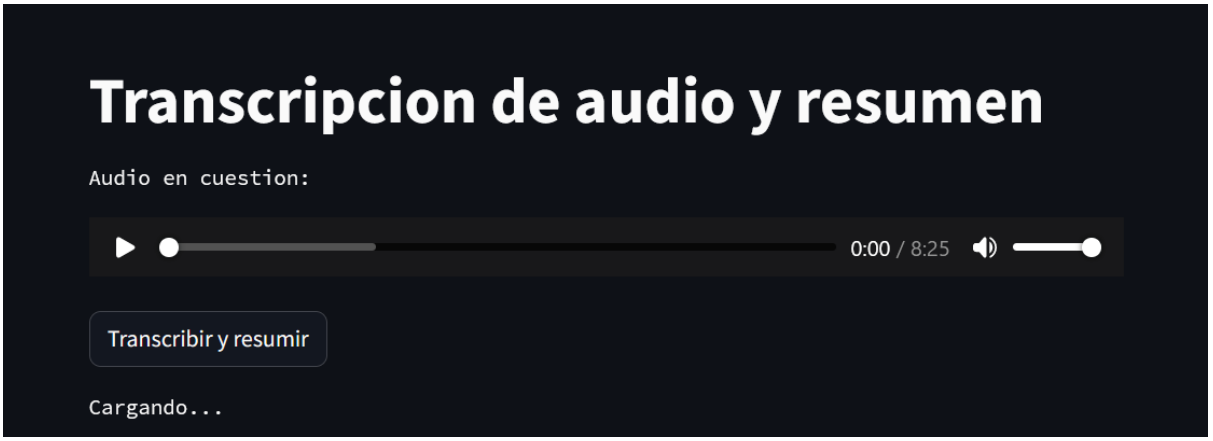
Este funciona de la misma manera que el archivo antes descrito. Sin embargo esta utiliza la librería de Streamlit para beneficio de poder tener un visualización de este. A continuación le muestro un ejemplo del flujo de esta página.

Flujo de la interfaz gráfica

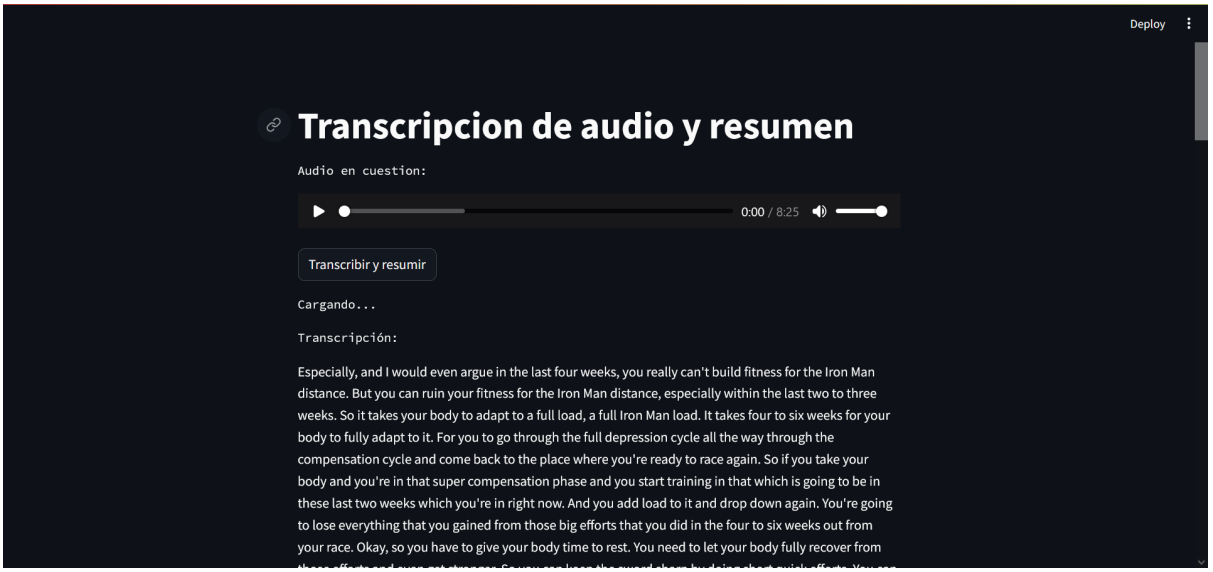
- **Inicio de la interfaz**. Se encuentra el título, y un reproductor con el audio a transcribir y resumir. Así como un botón para iniciar el proceso

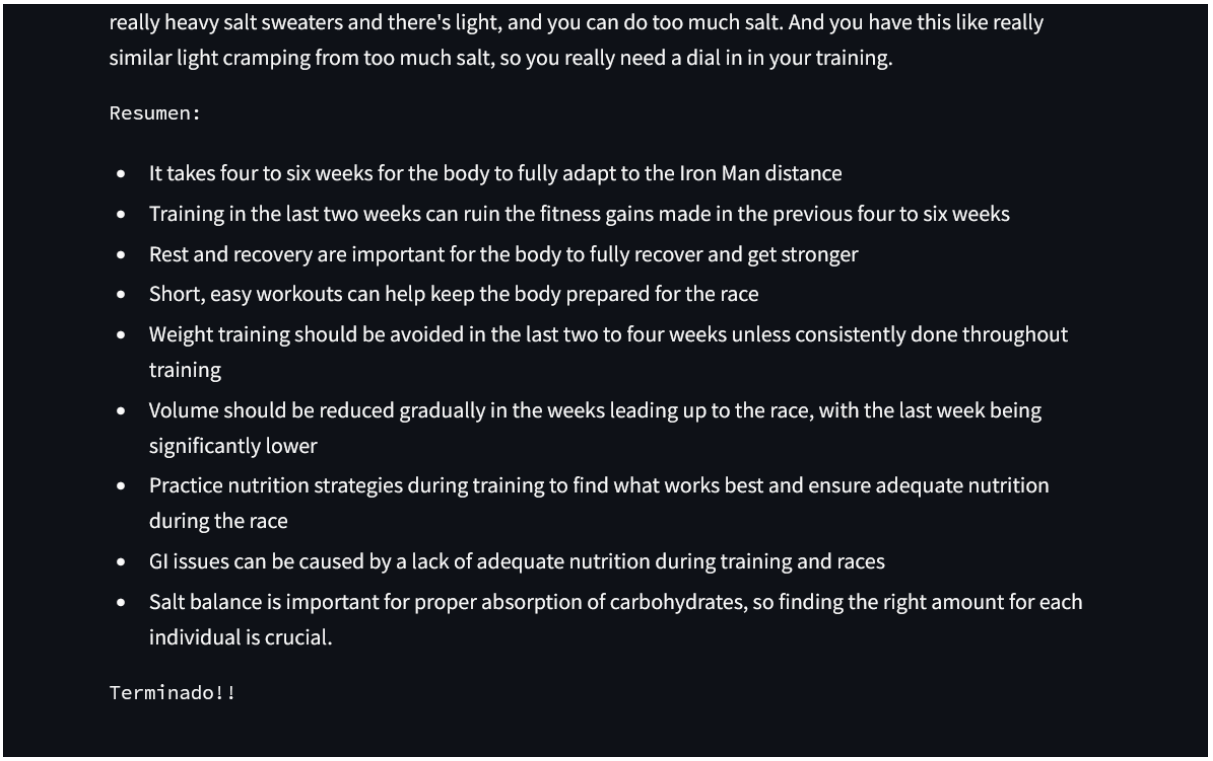


- **Carga de archivo**. Inmediatamente despues de presionar el botón, se inicia el proceso y se presenta un diálogo de “Cargando..”



- **Visualización.** Una vez estos acaben, se muestra la transcripción y la síntesis en conjunto





Transcripción completa:

All right, so we're almost ready to finish up here. I'm going to say, let's make a file cache to actually save these volumes. So let's say file cache, sim, something like that. I'm going to say go to job here. I'm going to put it in my sim folder and say it's going to be $0.0.0.0.0, which is the name of our node, which is file cache sim. That version 0, the current frame number, and then bgeo.sc is the file format for geometry. There you go. If you middle click it, you'll see the full thing there, including the frame number. So that's cool. It's 120, which we already know is right. We might want to do though, is realize we're getting a temperature field because we asked for it here. We're not going to use the temperature field to actually render with. You would use the temperature field if you're doing an explosion, because that would dictate the color of the fire. But we're not doing that here. Let's not import it. Let's not save it to disk. These files are going to be big enough as it is, no point of doing that. We don't have a rest field, so I don't even worry about that. The velocity field is trickier. This sim isn't moving that fast that the velocity blur will really matter that much. It might matter right at the source, but it's going to slow our sim down quite a bit. If you have a lower grade computer, I would say turn this off, both because it's going to take up a lot of disk space because it's a vector, meaning it's three times as much data as the density field is, which is only one dimensional. If you're not sure and you have the disk space to kill, then just then yes, just save it. Otherwise there you go. I'm going to do a cache right now of this and we're going to look at it. It should actually probably run pretty fast. But the actual sim we're going to run is going to be higher as in that. For that, we might walk away for and let it do its thing. Anyway, here it goes. We'll take a look at it in a minute. Okay. Click load to from disk. We scrub through it. There we go. We've got our whole sim there now. I guess technically this vign visualization should be after this so that we can make changes to it if we needed to from the cache. But you know, anyway. So there you go. Let's just run a little flip book. I mean, again, this is fairly low res. But let's just get an idea of it. Let's make sure that things aren't getting chopped off unexpectedly somewhere. The color all works. Everything works. It seems like it's going to be good. Yeah. So there you go. That looks nice. Good enough for me to say let's up-res this and then actually let that run. This is also a good time to say let's look at how big those files are. Oh, that was a mistake. Probably don't want to store these in your actual documents area. Usually a good idea to put this into something like data drive area. So a good thing we caught that before we did the

higher res thing. Apply to dini temp is where I'm going to do it. I'm going to make a new folder and call it, you know, line three, version two temp. So we'll put it in here instead. And then that'll be, I'll just copy this here. And then that'll be that. So I'll just save it to disk again real quick because it's already it's still a cache in memory because we've increased the cache from before. So it's not even resuming it. It's just rewriting it. So here you go. So these files are here. They're not that big. They're only like a hundred megs at the most. But of course, you know, every time we increase the resolution or rather, you

know, reduce the voxel size by half, everything is going to be doubled along each access. So twice as many voxels in the x-axis, twice as many voxels in the y-axis and twice as many in the z. Two times two times two is eight. Eight times as large. These things will become, this is going to go from being one hundred and fifteen megabytes to a gigabyte per frame. So just

keep that in mind. Then that's if I went to 0.05, which is again, reducing it by half is going to double it
along a react. I'm going to go ahead and say go even farther. 0.04 is going to make these files fairly large. These files are probably going to be probably about 1.2 gigs per frame by the end of it. But you know, you can buy terabyte drives now for a hundred bucks. So or less at this point, it is well worth it to have
that data there. Although just because you can doesn't mean you should either, with a very high quality volume, means you will need more steps in the rendering
process as the rays penetrate the volume to render them as we'll talk about, it will make it slower to render also. So it will be slower to sim and it will be slower to render as well. So keep that in mind. Like this sim is actually going to be quite fast even at this size. Even with a gigabyte cache file, you think they would take a long time to sim, but it's actually not because of how fast the Pyro Spark Cellar is. So but it will still take a long time to render. So just don't make, you could you could set to 0.02 and have 10 gigabyte cache files. And maybe you do have enough disk space for that, but they're going to be very slow to render. And they're going to look good like this anyway. So just saying don't don't do it just because you can. Okay. So I'm going to save this to disk and
this will probably take still I surprisingly short amount of time. But yeah, I'll take a few minutes. So let's see in a minute.